We Claim:

1. In a graphical modeling and execution environment, a method comprising the steps of:

providing a model view and an execution list view of a model being executed, said model view showing a plurality of components of said model, said execution list view showing an execution list depicting the execution order of methods called during the execution of a time step of said model, said model view interfaced with a debugger; and

indicating visually the state of the execution list on said model view.

2. The method of claim 1, comprising the further step of:

displaying a visual indicator indicating an association between an executing block method and a calling block on said model view.

3. The method of claim 1, comprising the further step of:

displaying a visual indicator indicating an association between a currently executing system method and a subsystem block owner of said currently executing system method on said model view.

4. The method of claim 1, comprising the further steps of:

creating a visual representation of a model component not previously displayed in said model view, said model component calling a method; and

displaying a visual indicator indicating an association between the visual representation of the model component not previously displayed and the method called by the model component.

5. The method of claim 1, comprising the further steps of:

extending a visual indicator from an originating point to a first called method depicted in said model view; and

extending sequentially said visual indicator to at least one of each subsequently called method depicted in said model view and a virtual subsystem in said model view during a time step in said execution.

6. The method of claim 5, comprising the further step of:

indicating the type of method executing in said model view.

7. The method of claim 6 wherein said indication is a visual indication.

8. The method of claim 7 wherein said visual indication is made by one of altering the color of a portion of a model component in said model view representing said method and inserting a geometric design in a model component displayed in said model view.

9. The method of claim 1 wherein a user sets visible breakpoints in said model view.

10. The method of claim 9 wherein said breakpoints are conditional breakpoints.

11. The method of claim 1, comprising the further step of:

arranging said execution list view to show the methods executed in a current time step in the execution of said model in a tree structure.

12. The method of claim 1 wherein a user sets visible breakpoints in said execution list view.

13. The method of claim 12 wherein said breakpoints are conditional breakpoints.

14. The method of claim 1, comprising the further step of:

setting at least one a trace point and a display point in at least one of said model view and said execution list view.

15. The method of claim 1, comprising the further steps of:

generating at least one of debugging data and profiling data during the execution of said model;

associating said at least one of debugging data and profiling data with at least one of said components of said model; and

visually indicating said associated data in said model view.

16. The method of claim 15 wherein said associated data includes solver data.

17. The method of claim 1, comprising the further steps of:

generating debugging data with said debugger during the execution of said model;

associating said debugging data with at least one of said components of said model; and

visually indicating said associated data in said execution list view.

18. The method of claim 17, comprising the further step of:

indicating visually in said execution list view the number of iterations of at least one of said plurality of model components during a time step in said execution.

19. The method of claim 1, comprising the further steps of:

selecting a user-set speed parameter via a control associated with said model view; and

executing said model in said model view based on the selected speed parameter.

20. The method of claim 1, comprising the further steps of:

selecting a user-set speed parameter via a control associated with said execution list view; and

executing said model in said execution list view based on the selected speed parameter.

21. The method of claim 1, comprising the further steps of:

receiving input from a user-controlled input device in said graphical modeling and execution environment, said input being interpreted by said graphical modeling and execution environment as a user-selected speed parameter; and

executing said model in said execution list view based on the selected speed parameter.

22. The method of claim 1, comprising the further steps of:

altering at least one of a connection between said model components and at least one of said model components; and

adjusting at least one of said execution list view and said model view to indicate the effects of said altering.

23. The method of claim 22 wherein said altering step includes at least one of the adding and removing of at least one of model components and a connection between said model components.

24. The method of claim 1, comprising the further step of:

displaying elements of the compiled state of said model in said model view.

25. The method of claim 1, comprising the further step of:

displaying debug information from said debugger to a user in said model view as a tool tip over a component of said model in response to user input .

26. The method of claim 25 wherein the displayed information indicates a signal value of a signal line in said model view.

27. The method of claim 25 wherein the displayed information is made persistent in said model view.

28. The method of claim 27 wherein said displayed information is updated in response to the execution of said model.

29. The method of claim 1, comprising the further step of:

displaying debug information from said debugger to a user in said execution list view as a tool tip in response to the movement of a pointing device in said execution list view over a component of said model associated with said debug information.

30. The method of claim 29 wherein the displayed information is made persistent in said execution list view.

31. The method of claim 30 wherein said displayed information is updated in response to the execution of said model.

32. The method of claim 1, comprising the further step of:

filtering the displayed execution list of methods in said execution list view so that only methods satisfying a user-specified criteria are displayed.

33. The method of claim 1, comprising the further steps of:

creating a record for each unique method invocation; and

displaying data associated with said unique method invocations as they are called.

34. The method of claim 33, comprising the further step of:

anchoring said record to a block owner of said unique method invocation in said model view.

35. The method of claim 33, comprising the further step of:

displaying the calling of said unique method invocation with varying degrees of intensity representive of the frequency of the invocation.

36. The method of claim 33, comprising the further step of:

creating a unique method invocation for an execution exception event.

37. The method of claim 1 wherein a user sets non-visible breakpoints in at least one of said model view and said execution list view.

38. The method of claim 1 wherein at least one of a set of debugging data and a set of profiling data are displayed to a user in a separate view.

39. A medium for use in a graphical modeling and execution environment on an electronic device, said medium holding instructions executable on said electronic device for performing a method , said method comprising the steps of:

providing a model view and an execution list view of a model being executed , said model view showing a plurality of components of said model, said execution list view showing an execution list depicting the execution order of methods called during the execution of a time step of said model, said model view interfaced with a debugger; and

indicating visually the state of the execution list on said model view.

40. The medium of claim 39, wherein said method comprises the further step of:

displaying a visual indicator indicating an association between an executing block method and a calling block on said model view.

41. The medium of claim 39, wherein said method comprises the further step of:

displaying a visual indicator indicating an association between a currently executing system method and a subsystem block owner of said currently executing system method on said model view.

42. The medium of claim 39, wherein said method comprises the further steps of:

extending a visual indicator from an originating point to a first called method depicted in said model view; and

extending sequentially said visual indicator to each subsequently called method depicted in said model view during a time step in said execution.

43. The medium of claim 42, wherein said visual indicator is extended to a virtual subsystem depicted in said model view.

44. The medium of claim 42, wherein said method comprises the further step of:

indicating the type of method executing in said model view.

45. The medium of claim 44 wherein said indication is a visual indication.

46. The medium of claim 45 wherein said visual indication is made by one of altering the color of a portion of a model component in said model view representing said method and inserting a geometric design in a model component displayed in said model view.

47. The medium of claim 39 wherein a user sets visible breakpoints in said model view.

48. The medium of claim 47 wherein said breakpoints are conditional breakpoints.

49. The medium of claim 39, wherein said method comprises the further step of:

arranging said execution list view to show the methods executed in a current time step in the execution of said model in a tree structure.

50. The medium of claim 39 wherein a user sets visible breakpoints in said execution list view.

51. The medium of claim 50 wherein said breakpoints are conditional breakpoints.

52. The medium of claim 39, wherein said method comprises the further step of:

setting at least one a trace point and a display point in at least one of said model view and said execution list view.

53. The medium of claim 39, wherein said method comprises the further steps of:

generating at least one of debugging data and profiling data with said debugger during the execution of said model;

associating at least one of said debugging data and profiling data with at least one of said components of said model; and

visually indicating said associated data to a user in said model view.

54. The medium of claim 53 wherein said associated data includes solver data.

55. The medium of claim 39, wherein said method comprises the further steps of:

generating debugging data with said debugger during the execution of said model;

associating said debugging data with at least one of said components of said model; and

visually indicating said associated data to a user in said execution list view.

56. The medium of claim 55, wherein said method comprises the further step of:

indicating visually in said execution list view the number of iterations of at least one of said plurality of model components during a time step in said execution.

57. The medium of claim 39, wherein said method comprises the further steps of:

selecting a user-set speed parameter via a control associated with said model view; and

executing said model in said model view based on the selected speed parameter.

58. The medium of claim 39, wherein said method comprises the further steps of:

selecting a user-set speed parameter via a control associated with said execution list view; and

executing said model in said execution list view based on the selected speed parameter.

59. The medium of claim 39, comprising the further steps of:

receiving input from a user-controlled input device in said graphical modeling and execution environment, said input being interpreteted by said graphical modeling and execution environment as a user-selected speed parameter ; and

executing said model in said execution list view based on the selected speed parameter.

60. The medium of claim 39, wherein said method comprises the further steps of:

altering at least one of a connection between said model components and at least one of said model components; and

adjusting at least one of said execution list view and said model view to indicate the effects of said altering.

61. The medium of claim 60, wherein said altering step includes at least one of the adding and removing of at least one of model components and a connection between said model components.

62. The medium of claim 39 wherein said method comprises the further step of:

displaying elements of the compiled state of said model in said model view.

63. The medium of claim 39, wherein said method comprises the further step of:

displaying debug information from said debugger to a user in said model view as a tool tip over a component of said model in response to user input. .

64. The medium of claim 63 wherein the displayed information indicates a signal value of a signal line in said model view.

65. The medium of claim 63 wherein the displayed information is made persistent in said model view.

66. The medium of claim 65 wherein said displayed information is updated in response to the execution of said model.

67. The medium of claim 39, wherein said method comprises the further step of:

displaying debug information from said debugger to a user in said execution list view as a tool tip in response to the movement of a pointing device in said execution list view over a component of said model associated with said debug information.

68. The medium of claim 67 wherein the displayed information is made persistent in said execution list view.

69. The medium of claim 68 wherein said displayed information is updated in response to the execution of said model.

70. The medium of claim 39, wherein said method comprises the further step of:

filtering the displayed execution list of methods in said execution list view so that only methods satisfying a user-specified criteria are displayed.

71. The medium of claim 39, wherein said method comprises the further steps of:

creating a record for each unique method invocation; and

displaying data associated with one of said unique method invocations as said unique method invocation is called.

72. The medium of claim 71, wherein said method comprises the further step of:

anchoring said record to a block owner of said unique method invocation in said model view.

73. medium of claim 71, wherein said method comprises the further step of:

displaying the calling of said unique method invocation with varying degrees of intensity representive of the frequency of the invocation.

74. The medium of claim 71, wherein said method comprises the further step of:

creating a unique method invocation for an execution exception event.

75. The medium of claim 39 wherein a user sets non-visible breakpoints in at least one of said model view and said execution list view.

76. The medium of claim 39 wherein at least one of a set of debugging data and a set of profiling data are displayed to a user in a separate view.

77. In a graphical design environment, a system comprising:

a debugger, said debugger gathering debug information from the simulation of a model in said graphical design environment;

a model view, said model view displaying a plurality of components of a model and being interfaced with said debugger; and

an execution list view, said execution list view displaying an execution list depicting an execution order of methods called during the execution of a time step of said model, said execution list view state being visually represented on said model view, said execution list view being generated by said debugger.

78. The system of claim 77, comprising further:

a visual indicator indicating a currently executing method on said model view.

79. The system of claim 78 wherein said visual indicator sequentially extends said indicator to denote said execution order of methods on said model view.

80. The system of claim 77 wherein a user is able to set at least one of breakpoints, conditional breakpoints, display points and trace points on said model view.

81. The system of claim 77 wherein a user is able to set at least one of breakpoints, conditional breakpoints, display points and trace points on said execution list view.

82. The system of claim 77 wherein a visual indicator is used to indicate the type of executing method displayed in said model view.

83. The system of claim 82 wherein said visual indicator is one of color and a geometric pattern.